DS

تراكيب البيانات
DATA STRUCTURES

ITGS220

موضوع الدراسة : تراكيب البيانات الخطية Linear data structures

1. القـــوائـم Lists .
   - (الطابور الخطي Linear Queue)

أستاذة المادة
أ. وفاء حسين المصباحي

DS

المواضيع التي سيتم دراستها في مقرر
تراكيب البيانات

1. مقدمة تمهيدية Introductory Review .
2. تراكيب البيانات الخطية Linear data structures .
3. الترتيب Sorting .
4. تراكيب البيانات الغير خطية Non-Linear data structures .

2. تراكيب البيانات الخطية Linear data structures .

2. تراكيب البيانات الخطية Linear data structures :

- القـــــوائـــم Lists .
  - الطابور الخطي Linear Queue .
    - العمليات التي تجرى على الطابور الخطي The operations of Linear Queue .
    - التطبيق للطابور Implementing of Queue .
    - برنامج بلغة السي لتطبيق الطابور الخطي بواسطة المصفوفة

Code with C language for implementing a linear queue with an array .

▪ الطابور الخطي Linear Queue :

التعريف Definition :

- **الطابـور الخطي** Linear queue **هو قائمة خطية** linear list **من العناصر** items ، **يستخـدم في مؤشرين** two pointers :

- **المؤشر الأول** The first pointer : **يدعي المؤشر الخلفـــي** REAR **يستخدم لإدخال** insert **عنصر واحد في الطابور** queue .

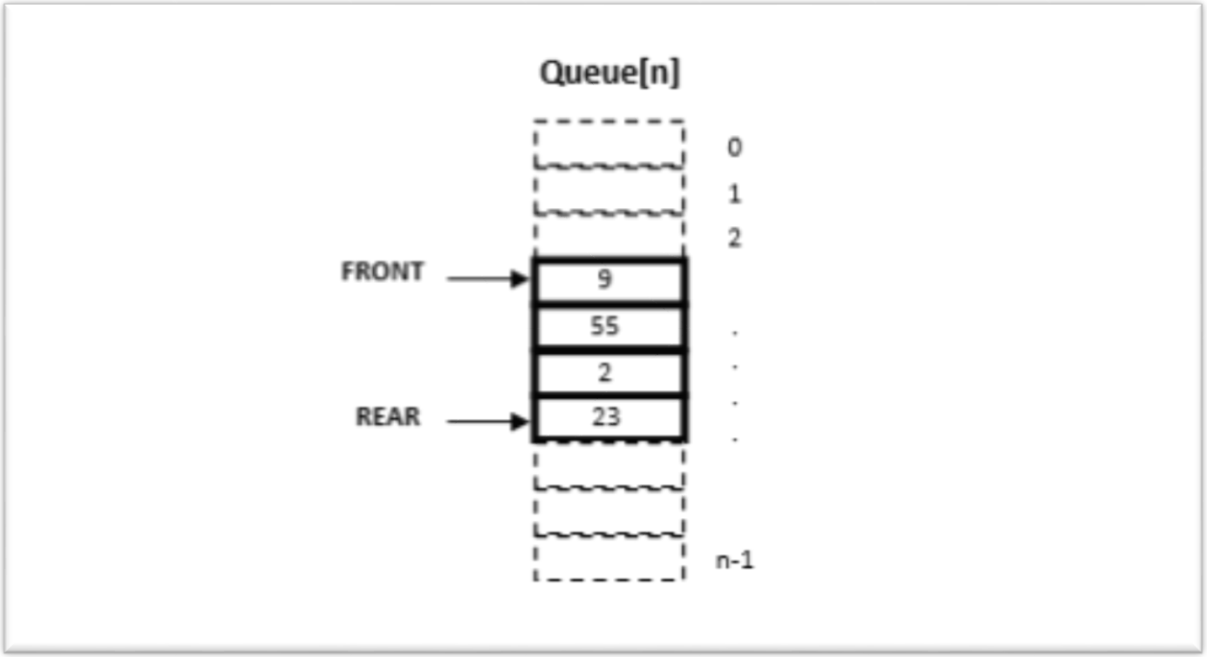
- **المؤشر الثاني** The second pointer : **يدعي المؤشر الأمامـــي** FRONT **يستخدم لإلغاء** delete **عنصر واحد من الطابور** queue .

لذلك، الطابور الخطي Linear queue **يعتبر قائمة متغيرة** Dynamic List .

▪ الطابور الخطي Linear Queue :

▪ **الطابور الخطي** Linear queue **يعتبر** First-In-First-Out (FIFO) data structure ، **العنصر الأول المضاف للطابور** queue **هو أول عنصر يتم إزالته من الطابور** queue . **وهذا يكون مكافئ لمتطلبات تلك العناصر ، كل العناصر التي يتم إضافتها أولاً يتم إزالتها أولاً.**

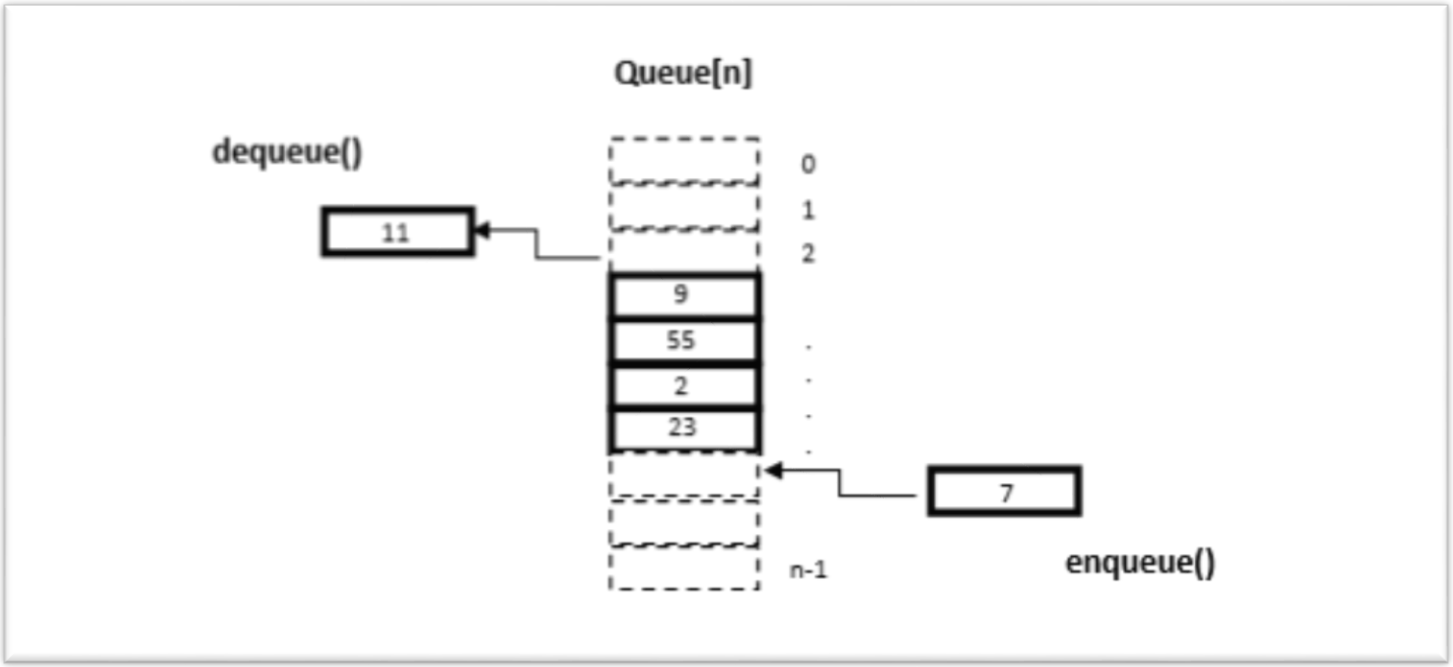
FIFO → (First in, First out)



العمليات التي تجرى على الطابور The operations of queue :

العمليات الرئيسية التي تنجز على الطابور queue :

1. enqueue(item) : إدخال insert عنصر في الطابور queue من مؤشر rear .
2. dequeue() : إلغاء delete وإرجاع قيمة العنصر الذي يأشر له المؤشر front .



تطبيقات الطابور Application of Queue :

1. تقنية Round Robin technique لجدولة المعالج processor scheduling تكون مطبقة باستخدام تركيبة بيانات الطابور the queue data structure .
2. كل خدمات الزبائن مثل (نظام الحجز في الطيران) البرمجيات المصممة تستخدم تركيبة بيانات الطابور the queue data structure لتخزين معلومات الزبائن customers information .
3. Printer server routines تكون مطبقة باستخدام تركيبة بيانات الطابور the queue data structure .

أنواع الطابور Types of Queue :

الطابور queue **يمكن أن يكون :**

1. **طابور خطي** Linear queue .
2. **طابور دائري** Circular queue .

التطبيق للطابور Implementation of queue :

في أغلب لغات البرمجة العالية المستوى high level languages ، الطابور queue يمكن أن يطبق بسهولة.

```
CREATE  Q[n] , FRONT ← -1 , REAR ←  -1
```

```
INSERT ( Q[n] , REAR , x )
IF  REAR = n-1 , THEN  "Queue Full"
IF  REAR = -1 & FRONT=-1 , THEN  FRONT ← 0
REAR  ←  REAR + 1
Q[REAR] ←  x
END
```

```
DELETE ( Q[n] , FRONT, REAR )
IF  FRONT = REAR , THEN  FRONT = REAR =-1  "Queue Empty"
FRONT ←  FRONT + 1
END
```

برنامج بلغة السي لتطبيق الطابور الخطي بواسطة المصفوفة

Code with C language for implementing a Linear Queue with an array :

برنامج بلغة السي لتطبيق الطابور الخطي بواسطة المصفوفة

Code with C language for implementing a Linear Queue with an array :

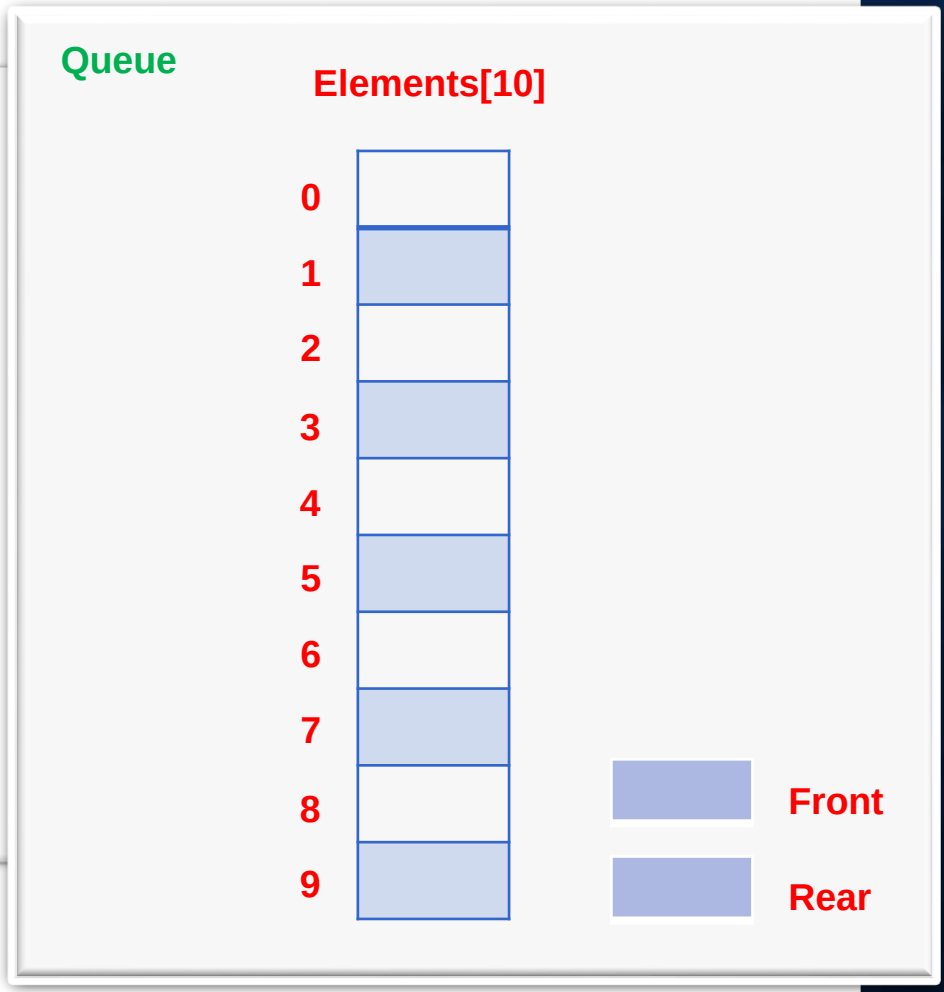
العمليات على الطابور الخطي Operations on Linear Queue :

- createqueue(q) : لتكوين q كطابور فاضي queue empty .
- enqueue(q,i) : لإدخال عنصر i في q .
- dequeue(q) : للوصول وإزالة عنصر من الطابور queue q .
- peek(q) : للوصول إلى أول عنصر في الطابور queue q بدون إزالته.

تمثيل الطابور Queue Representation :

نستخدم الإعلانات التالية لتمثيل الطابور queue :

```
/*Define a queue of capacity 10*/
#define CAPACITY 10
typedef struct
{
    int front;
    int rear;
    int elements[CAPACITY];
} queue;
queue *q;
```

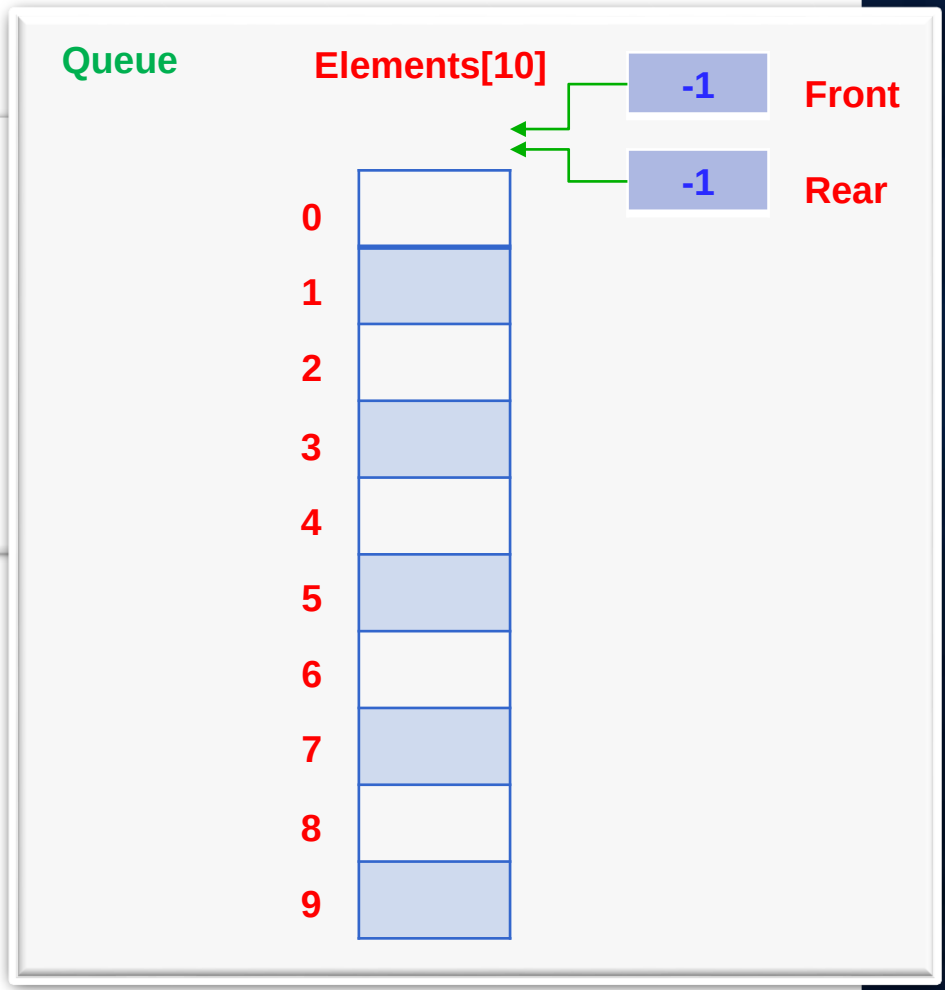


باستخدام هذه الإعلانات، نستطيع إنجاز العمليات المختلفة على الطابور queue .

تكوين الطابور الفاضي Creating an Empty Queue :

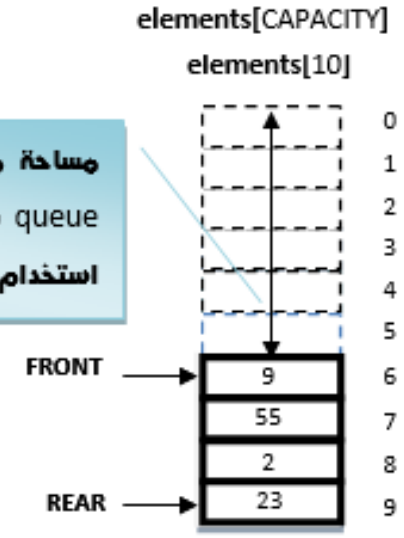
قبل أن نستخدم الطابور queue ، يجب أن يتم تكوينه . ويتم تكوين الطابور queue وذلك بتخصيص قيمة 1- لكل من المتغيرين front و rear . نلاحظ بأن المدى المسموح للمصفوفة هو من 0 إلى CAPACITY-1 .

```
void createqueue(queue *q)
{
    q->front=q->rear=-1;
}
```

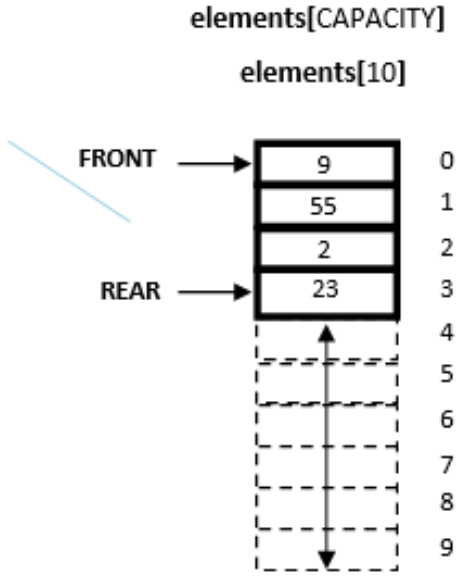


إنجاز عملية enqueue على الطابور الخطي Performing the enqueue Operation on a Linear Queue :

مساحة مزعجة، حيث لا يمكن إضافة عنصر item في هذه المساحة من الطابور queue بالرغم من وجود مكان فاضي في الطابور queue ، مما يؤدي إلى سوء استخدام الذاكرة.



وفي هذه الحالة نقوم بإزاحة FRONT ، REAR إذا كان يوجد أماكن فاضيه.



```
void enqueue(queue *q, int value)
  {
    int i;
    if(q->front==-1 && q->rear==-1)          /* Queue is empty */
       q->front=0;
    else

     if ((q->rear == CAPACITY-1) && (q->front > 0))
      {
       for(i=q->front;i<=q->rear;i++)
          q->elements[i-q->front]=q->elements[i];
       q->rear=q->rear-q->front;
       q->front=0;
      }
     else

       if ((q->rear == CAPACITY-1) && (q->front == 0)) /*Queue is full*/
        {  printf("Queue is full");
           exit(0);
        }

    q->rear++;
    q->elements[q->rear]=value;
  }
```

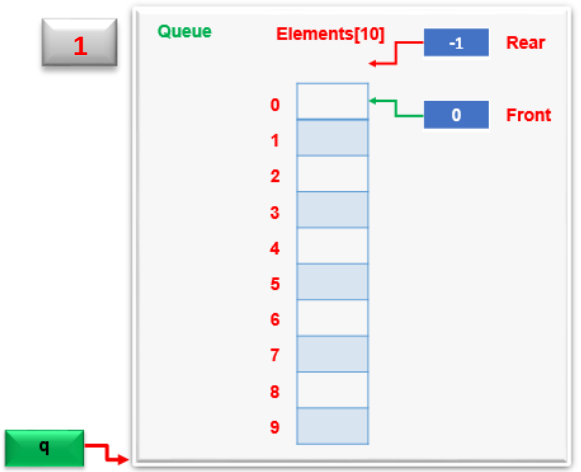
1 الطابور فاضي

2 إزاحة العناصر

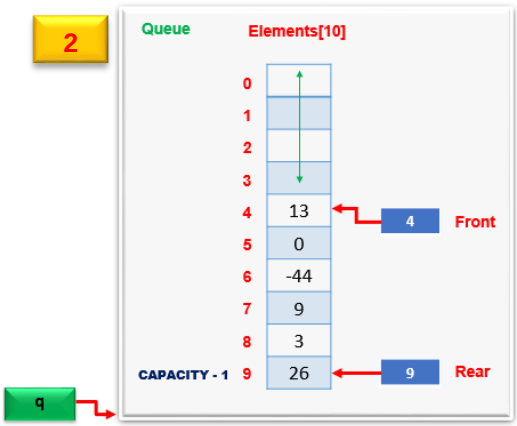
3 الطابور ممتلئ

4 إضافة عنصر

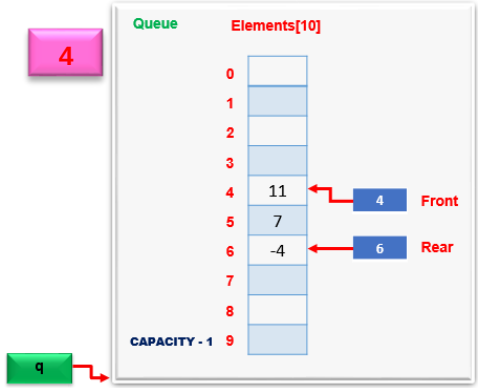
1



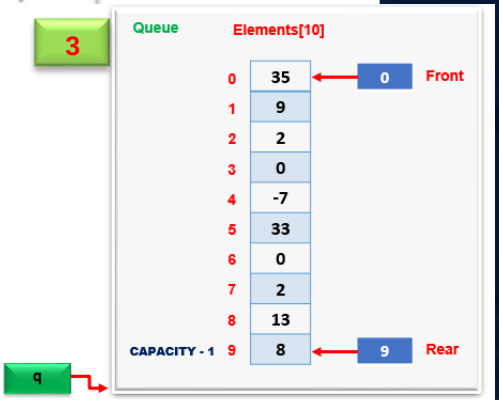
2



4



3



إنجاز عملية dequeue على الطابور الخطي Performing the dequeue Operation on a Linear Queue :

```
void dequeue(queue *q)
 {
  if(q->front == q->rear)
    q->front=q->rear=-1;
  else
    q->front++;
 }
```

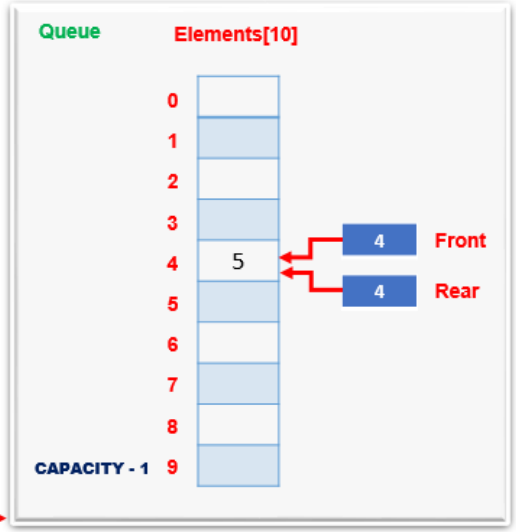
1

يوجد عنصر واحد داخل الطابور

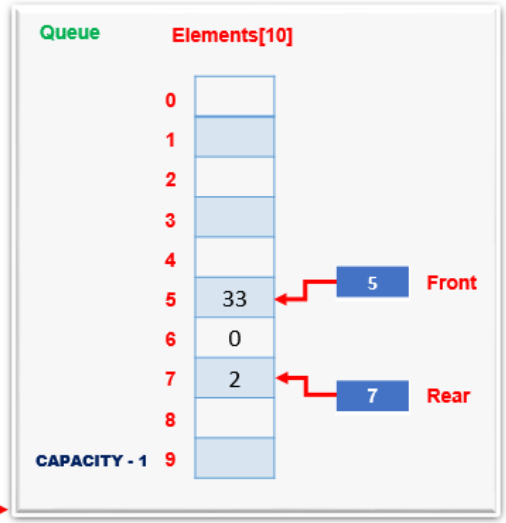
2

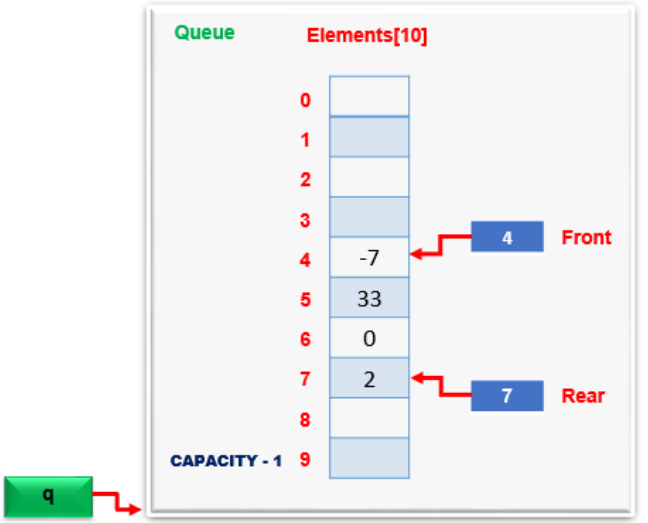
الغاء عنصر

1



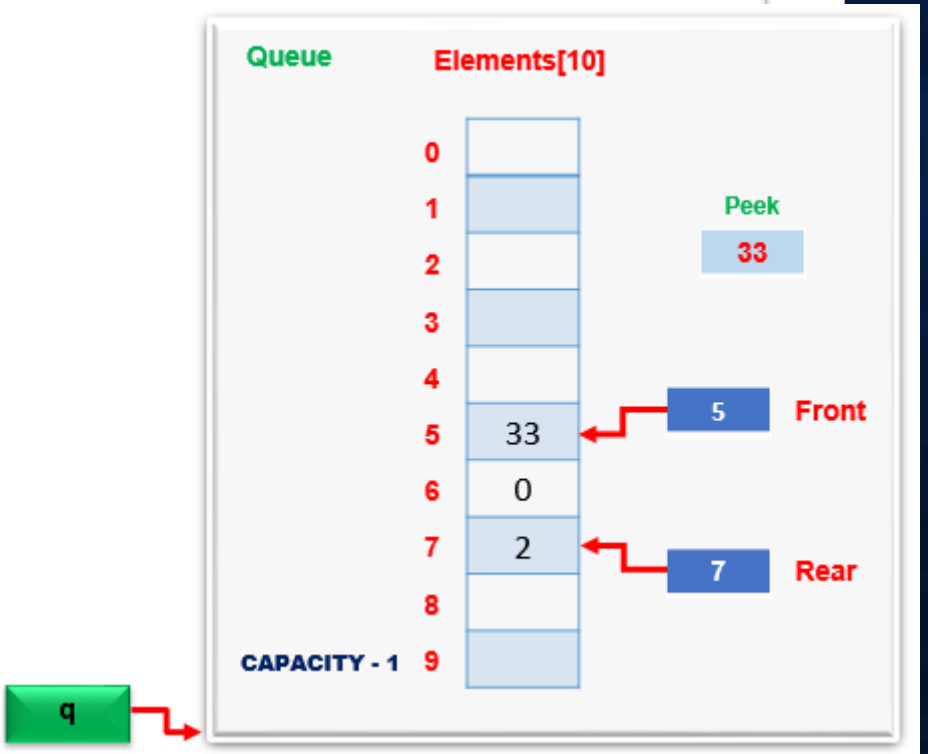
2





الوصول إلى العنصر الأول في الطابور Accessing to the first Element :

```
int peek(queue *q)
{
    return(q->elements[q->front]);
}
```



THANK YOU